SpiNNcloud

Event-based Backpropagation on SpiNNaker2: On-chip and hybrid training using EventProp algorithm

Gabriel Béna

Imperial College London & SpiNNcloud Systems

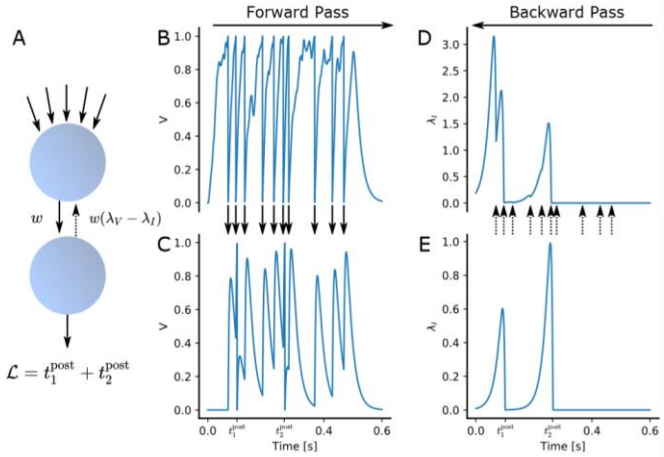
Introduction and Context

- We want to train networks on-chip.
  - Reduce energy footprint of training.
  - Key for unlocking true potential of embedded intelligent agents ?
  - Requires retaining sparsity of event-based communications during training
- We also might want to use **exact** gradients. Usually we use surrogates, but what is the impact of this ?
  - It has been suggested that surrogate are robust.
  - Impact of surrogate vs exact still not fully understood.
- But : on-chip learning is hard !

The Remarkable Robustness of Surrogate Gradient Learning
Zenke, Vogels, 2020

The EventProp Algorithm

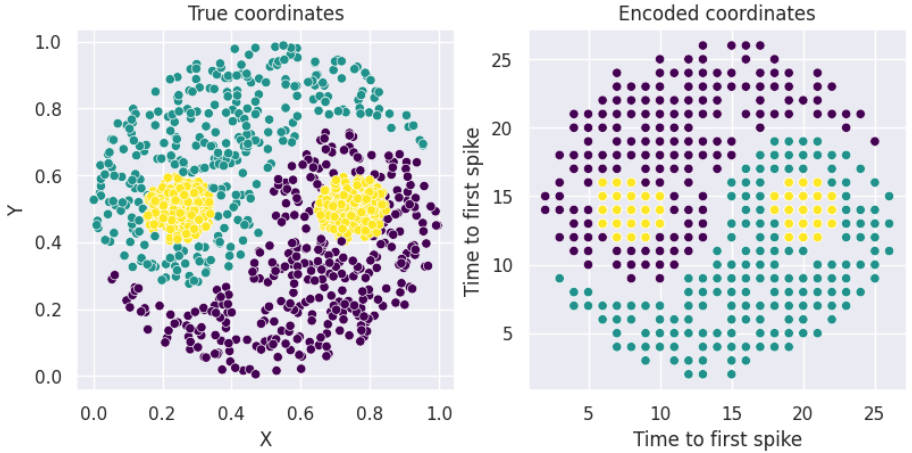
- Exact gradient computation that is **purely event-based** (sparse vs dense sampling for surrogate), which is good for hardware.
- The idea is that we can get gradients by computing the *adjoint system* of an SNN:
  - Basically, after the simulation has run in forward time, we run it again in reverse-time according to a new set of dynamics.
  - This is not more expensive than forward pass.
  - Using activity of reverse-time mode, we recover exact gradients!



Event-Based Backpropagation can compute Exact Gradients for Spiking Neural Networks
Wunderlich, Pelhe 2022

Yin-Yang Dataset

- Small scale: easy to fit on-chip.
- Not linearly separable.
- Challenging for shallow network to learn
  - This in turns help us validate gradient propagation in multi-layer network.



The Yin-Yang dataset
Kriener et al, 2021

SpiNNcloud

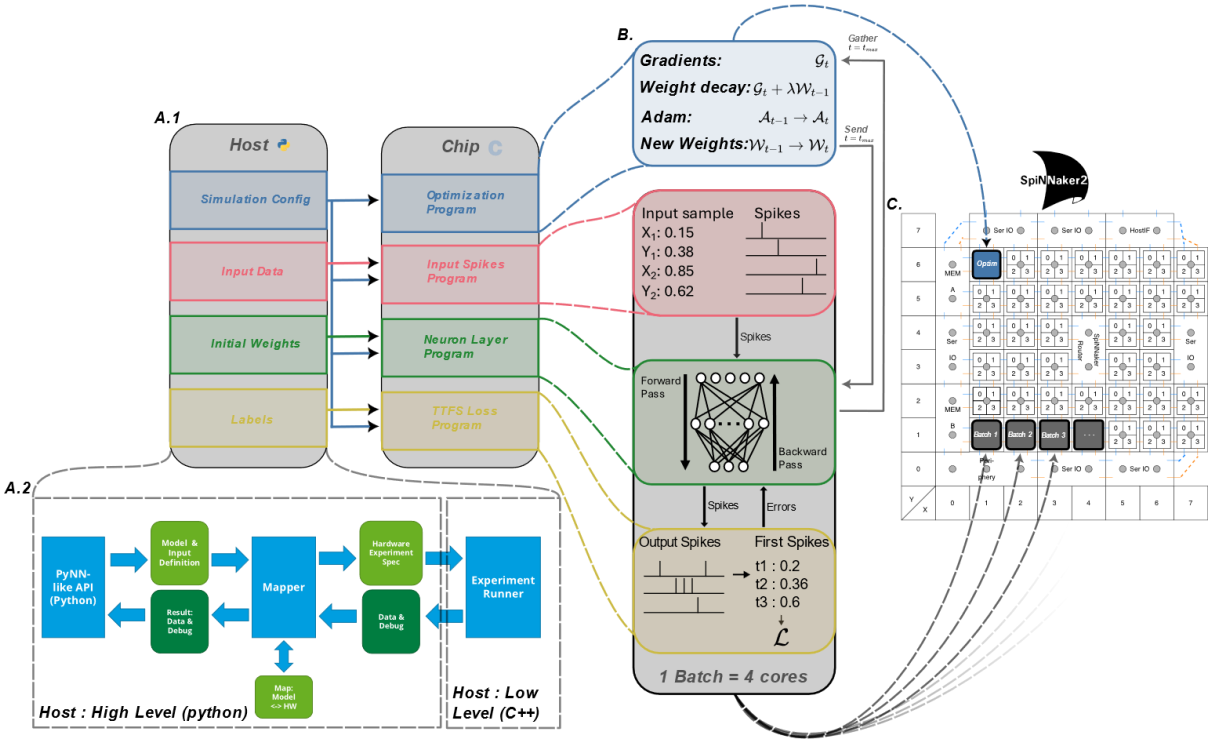
On-Chip Implementation

- One PE per dedicated population of neurons:
  - Input Spikes
  - Neuron layer 1
  - Neuron layer 2
  - Loss Computation

  } 1 Quad-PE

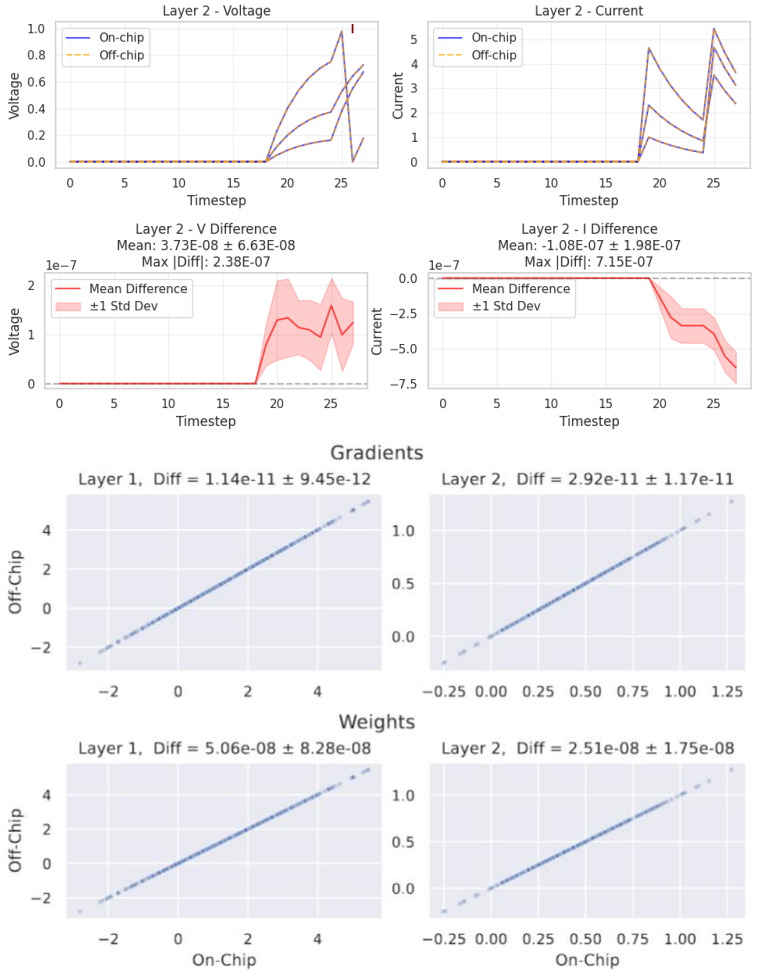
  } 1 Batch

- 1 global optimizer (Adam) PE.
  - Synchronize sample processing.
  - Gather gradients across batches.
  - Compute new weights.
  - Dispatch new weights before starting new sample processing.



Off- Chip Simulator

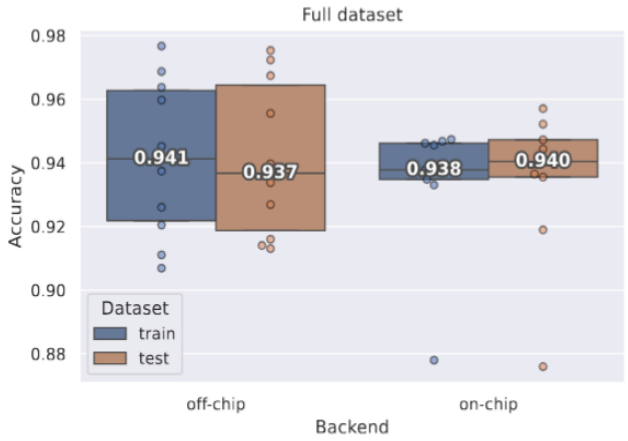
- Fully identical simulator.
- Purpose:
  - Efficient prototyping
  - Convenient debugging
  - Parameter search and optimization
- Implementation details:
  - Custom PyTorch package
  - Matches on-chip simulation behaviour
  - Available at pytorch-eventprop (open-source)
- Bayesian hyperparameter optimization:
  - "Weights and Biases" software package
  - Optimized parameters were then used for on-chip implementation
- Hybrid training (more later).

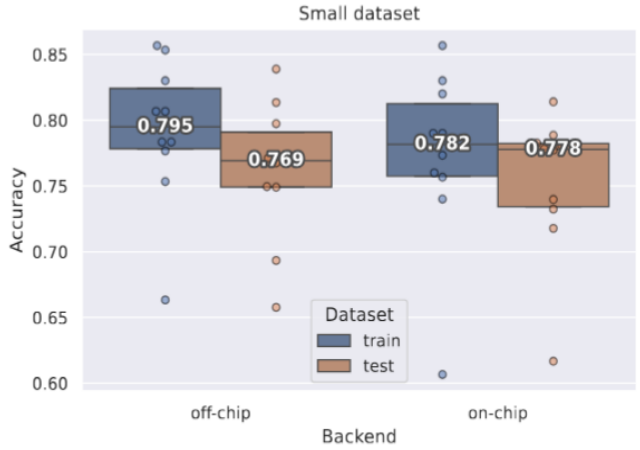


Training Results

- Full Dataset
  - 5000 training samples
  - Batch Processing
  - 20 epochs
- “Online” Dataset
  - 300 samples
  - Batch 1
  - Single pass trough the data

- Close match with off-chip
- Demonstrate the viability of training on-chip





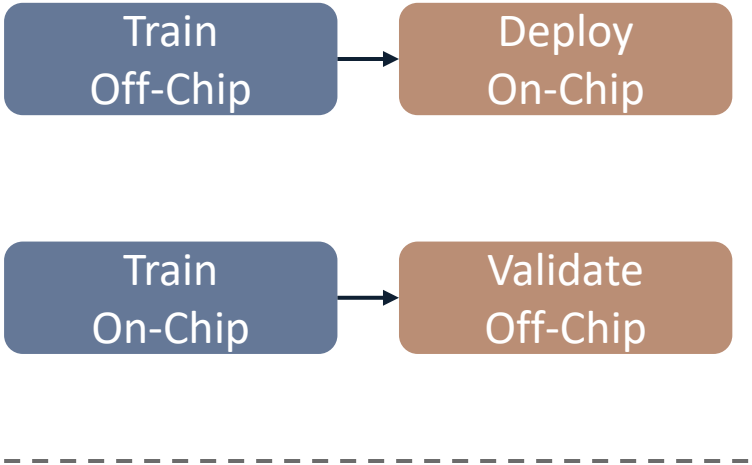
On-chip Time	Power	Energy (p. inference)
61 ms (30 + 30 + 1)	0.45 W	~27 mJ (1.22)

Energy Profiling Results

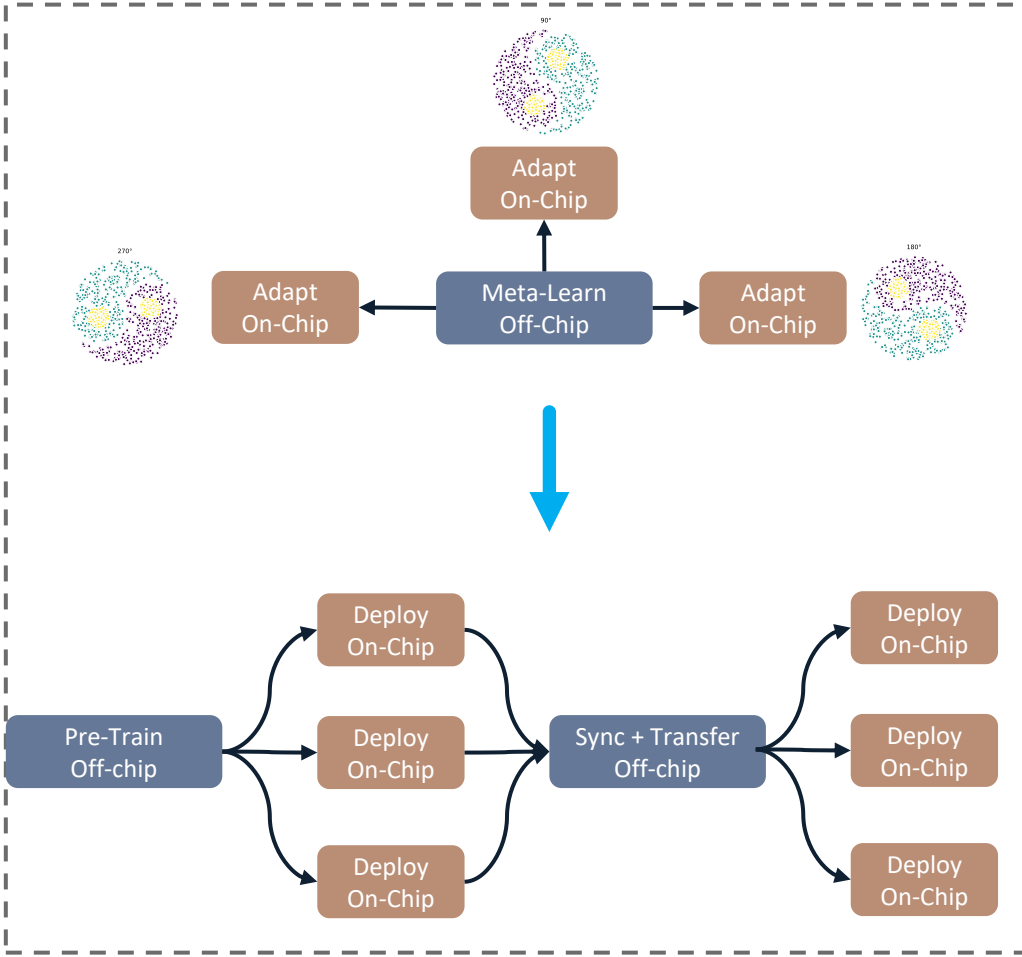
Forwards + Backward + Update

Batch Size 22

Hybrid Worflows



Implemented



Future Work

Meta-learning spiking neural networks with surrogate gradient descent
Stewart, Neftci 2022

Conclusions, Future Work

- Training in-chip is possible…
- …But scaling EventProp might be hard.
  - DRAM now available
  - Bigger networks and datasets
- We can utilize NIR for better On-chip <-> Off-chip.
- Super excited about Jaxsnn !
  - JAX + NIR + MAML should be a match for effective meta-learning + adaptation ?
- →Let's get "best of both world" via hybrid learning approaches !

Come to SpiNNaker2 Tutorial !!

SpiNNcloud

Thank you
for your attention

Thanks to Timo Wunderlich, Bernhard Vogginger and SpiNNcloud Colleagues !